Énumération des occurrences d'une chronique

Thomas Guyet*, Philippe Besnard**, Ahmed Samet***,
Nasreddine Ben Salha***, Nicolas Lachiche****

*AGROCAMPUS-OUEST/IRISA UMR 6074 IRISA
**CNRS/IRIT
***INSA Strasbourg/Laboratoire ICube
****Université de Strasbourg/Laboratoire ICube

Résumé. Une chronique est une représentation du comportement d'un système dynamique formalisée comme un ensemble d'événements liés par des contraintes temporelles. Dans cet article, nous nous intéressons à l'énumération des occurrences exactes ou approchées d'une chronique en comparant différentes stratégies de reconnaissances. Nous proposons un algorithme de reconnaissance de chroniques avec différentes heuristiques et sa version approchée. L'efficacité de ces différentes approches est évaluée sur des données synthétiques.

1 Introduction

Pour un grand nombre de contextes applicatifs, des données de nature séquentielle sont collectées. Par exemple, il peut s'agir d'une séquence de soins dont un patient a bénéficié, d'une séquence d'achats qu'a effectué un client dans un supermarché ou encore de *log* de fonctionnement d'une machine industrielle. Il est intéressant d'explorer ces données pour extraire de l'information sur les situations qui se sont effectivement déroulées.

L'approche présentée dans cet article cherche à interroger une base de séquences avec un motif défini par l'utilisateur. Un motif correspond à une situation dont l'utilisateur cherche les occurrences. Généralement un motif tient compte de l'ordre des événements mais rarement de l'intervalle de temps entre ces évènements. Pourtant, cette dimension temporelle peut contenir une information pertinente. Par exemple, les occurrences d'une situation où une hospitalisation survient après qu'une personne ait fait une chute est moins informatif qu'une situation correspondant à une hospitalisation survenue moins de 24h après la chute.

Dans la littérature, certaines approches retrouvent les occurrences d'une situation complexe dans des séquences d'événements en prenant en compte la dimension temporelle. Les méthodes de raisonnement temporel ou les logiques d'action répondent en particulier au problème de la reconnaissance de situations (Ulus et Maler, 2018; Artikis et al., 2012). Ces approches sont très expressives mais manquent d'efficacité. Les méthodes efficaces de reconnaissance font appel à des méthodes d'indexation (Kostakis et Papapetrou, 2017), mais sont peu expressives. Dans le domaine de la surveillance automatique, les approches basées sur les automates temporisés (Alur et Dill, 1994; Carle et al., 2011) permettent de spécifier des comportements avec des modèles formels. Ces représentations peuvent être difficiles à appréhender par des

utilisateurs et les algorithmes sont dédiés à la reconnaissance en ligne. Finalement, la fonctionnalité de recherche de motifs temporels est souvent rencontrée dans des outils d'analyse interactive de données temporelles (Monroe et al., 2013; Barazzutti et al., 2016).

Dans tous ces travaux, la reconnaissance est menée de manière exacte. Ainsi, même un écart infime sur les dates des événements ne permet pas d'informer l'utilisateur de la présence d'une occurrence proche du motif recherché. Pourtant, l'intérêt de telles occurrences est double : 1) elles permettent à un utilisateur de retrouver des occurrences d'un motif dont il ne peut pas spécifier avec certitude les contraintes temporelles, 2) cela permet d'identifier dans une séquence des occurrences qui seraient altérées par rapport au motif nominatif.

La contribution principale de cet article est la proposition d'un algorithme pour l'énumération des occurrences exactes et approchées d'une chronique dans une séquence.

2 Chroniques

Soit $\mathbb{E}$ un ensemble de types d'événements totalement ordonné. Cet ordre est dénoté $\leq_{\mathbb{E}}$. Un événement est un couple (e, t) où $e \in \mathbb{E}$ est un type d'événement et $t \in \mathbb{R}$ est une étiquette temporelle. Une collection ordonnée d'événements forme une séquence. Plus formellement, une séquence $\langle (e_1, t_1), (e_2, t_2), \ldots, (e_m, t_m) \rangle$ est une succession d'événements ordonnés par leur date puis leur type. On a alors pour tout couple $i, j \in [1..m], i < j \Leftrightarrow t_i \leq t_j$ et si $t_i = t_j$ alors $e_i \leq_{\mathbb{E}} e_j$. $|\boldsymbol{s}|$ dénote la longueur de la séquence $\boldsymbol{s}$.

Une chronique est une représentation du comportement d'un système dynamique comme un ensemble d'événements liés par des contraintes temporelles (Besnard et Guyet, 2019). Les premiers algorithmes d'énumération de chronique ont été proposés par Ghallab (1996) et ont des connexions fortes avec les réseaux de contraintes temporisés de Dechter et al. (1991).

Définition 1 (Chronique)**.** *Une **chronique** est un couple $(\mathcal{E}, \mathcal{T})$ tel que*

- *$\mathcal{E}$ est un multi-ensemble ordonné,* i.e.*, $\mathcal{E}$ est de la forme $\{\!\{e_1, \ldots, e_n\}\!\}$ (les répétitions sont permises) tel que $e_i \in \mathbb{E}$ pour $i = 1, \ldots, n$ et $e_1 \leq_{\mathbb{E}} \cdots \leq_{\mathbb{E}} e_n$;*
- *$\mathcal{T}$ est un ensemble de contraintes temporelles. Une contrainte temporelle est une expression de la forme $(e, o_e)[t^-, t^+](e', o_{e'})$ telle que toutes les conditions suivantes sont satisfaites : $e, e' \in \mathcal{E}$, $t^-, t^+ \in \overline{\mathbb{R}}$, $o_e, o_{e'} \in [n]$, $o_e < o_{e'}$, $e_{o_e} = e$ et $e_{o_{e'}} = e'$.*

La taille d'une chronique est la cardinalité de son multi-ensemble, ici n.

Définition 2 (Occurrence)**.** *Une occurrence d'une chronique $\mathcal{C} = (\{\!\{e_1, \ldots, e_n\}\!\}, \mathcal{T})$ dans une séquence $\boldsymbol{s} = \langle (s_1, t_1), \ldots, (s_m, t_m) \rangle$ est une liste de positions $(\epsilon_i)_{i \in [n]}$ telle que*

i. *$s_{\epsilon_i} = e_i$ pour tout $i \in [n]$, et*

ii. *$t_{\epsilon_{o'}} - t_{\epsilon_o} \in [l, u]$ pour toute contrainte temporelle $(e, o)[l, u](e', o') \in \mathcal{T}$*

*On dénote par $\mathcal{O}_{\mathcal{C}}(\boldsymbol{s})$ l'ensemble des occurrences d'une chronique $\mathcal{C}$ dans une séquence $\boldsymbol{s}$. Une chronique $\mathcal{C}$ **apparaît** dans une séquence $\boldsymbol{s}$, dénoté $\mathcal{C} \preceq \boldsymbol{s}$, si et seulement si il existe au moins une occurrence de $\mathcal{C}$ dans $\boldsymbol{s}$ (*i.e.*, $\mathcal{O}_{\mathcal{C}}(\boldsymbol{s})$ est non vide).*

On introduit à présent la notion d'occurrence approchée avec les (α, λ)-occurrences. Cette notion s'appuie sur l'idée que si les contraintes temporelles de la chronique sont suffisamment proches des délais d'une occurrence alors il faut reconnaître cette occurrence. Le paramètre $\alpha \in [0, 1]$ précise la tolérance admise sur les contraintes temporelles, tandis que le paramètre $\lambda \in \mathbb{R}_+^*$ module la mesure de similarité entre contrainte temporelle et occurrence.

Définition 3 ((σ,λ)-occurrence)**.** *Soit $\sigma \in [0,1]$, $\lambda \in \mathbb{R}_+^*$, une (σ,λ)-occurrence d'une chronique $\mathcal{C} = (\mathcal{E} = \{\!\{e_1,\ldots,e_n\}\!\}, \mathcal{T})$ dans une séquence $\boldsymbol{s} = \langle (s_1,t_1),\ldots,(s_m,t_m)\rangle$ est une liste de positions $(\epsilon_i)_{i\in[n]}$ telle que :*

i. $s_{\epsilon_i} = e_i$ *pour tout* $i \in [n]$*, et*

ii. $\sigma \leq \prod_{(e,o)[l,u](e',o')\in\mathcal{T}} \varsigma(t_{\epsilon_{o'}} - t_{\epsilon_o}, [l,u])$

avec

$$\varsigma(d,[l,u]) = \begin{cases} 1 & \text{si } d \in [l,u], \\ e^{-\lambda \min(|l-d|,|u-d|)} & \text{sinon.} \end{cases}$$

Lemme 1. *Pour $\lambda \in \mathbb{R}_+^*$, (ϵ_i) est une occurrence de $\mathcal{C}$ dans une séquence $\boldsymbol{s}$ ssi c'est une $(1,\lambda)$-occurrence de $\mathcal{C}$ dans $\boldsymbol{s}$.*

On introduit maintenant la notion de relaxation d'une chronique. La relaxation d'une chronique est une chronique dont les contraintes temporelles sont étendues.

Définition 4 (Relaxation de chronique)**.** *Soit $\alpha \in \mathbb{R}_+^*$. Pour une chronique $\mathcal{C} = (\mathcal{E},\mathcal{T})$, la α-relaxation de $\mathcal{C}$, notée $R_\alpha(\mathcal{C})$ est la chronique $(\mathcal{E},\mathcal{T}')$ telle que*

$$(e,o_e)[t^-,t^+](e',o_{e'}) \in \mathcal{T} \Leftrightarrow (e,o_e)[t^- - \alpha, t^+ + \alpha](e',o_{e'}) \in \mathcal{T}'$$

Lemme 2. *Pour tout $\lambda \in \mathbb{R}_+^*$ et tout $\sigma \in [0,1]$, si (ϵ_i) est une (σ,λ)-occurrence de $\mathcal{C}$ dans une séquence $\boldsymbol{s}$ alors (ϵ_i) est une occurrence de $R_{-\frac{1}{\lambda}\ln(\sigma)}(\mathcal{C})$ dans $\boldsymbol{s}$.*

3 Énumération des occurrences d'une chronique

L'algorithme 1 énumère toutes les occurrences d'une chronique dans une séquence. On verra ensuite comment adapter cet algorithme pour les (σ,λ)-occurrences. On propose une heuristique de parcours qui vise à identifier au plus tôt la non-reconnaissance d'une occurrence en explorant prioritairement les contraintes les plus dures : évaluation prioritaire des événements dont les types sont les moins fréquents et évaluation prioritaire des contraintes temporelles les plus restrictives.

Définition 5 (Région admissible)**.** *Étant données $\mathcal{C}$ de taille n et $\boldsymbol{s} = \langle (s_1,t_1),\ldots,(s_m,t_m)\rangle$, une région admissible est une séquence de n intervalles $\langle [r_1^-,r_1^+],\ldots,[r_n^-,r_n^+]\rangle$ telle qu'il existe (ϵ_i), une occurrence de $\mathcal{C}$ dans $\boldsymbol{s}$, avec $t_{\epsilon_i} \in [r_i^-,r_i^+]$ pour tout $i \in [n]$.*

Le principe est d'explorer un à un les événements du multi-ensemble en réduisant progressivement l'étendue des régions admissibles jusqu'à obtenir une région dont tous les intervalles sont des singletons. Une région admissible induit ainsi une occurrence de la chronique. L'ordre d'exploration des événements du multi-ensemble est donné par γ qui est construit dans l'algorithme par la fonction `next`. Nous reviendrons plus loin sur cette fonction.

L'algorithme 1 initie la récursion en identifiant toutes les occurrences du premier événement de la chronique dans la séquence. Lorsqu'une apparition du premier événement à traiter e_{γ_1} a été trouvée, une nouvelle région admissible est créée comme un intervalle singleton utilisant la date de l'événement dans la séquence. Puis, les lignes 8–13 propagent toutes les

Algorithme 1 : Énumération des occurrences d'une chronique.

Input : $\mathcal{C} = (\mathcal{E} = \{\!\{e_1, \ldots, e_n\}\!\}, \mathcal{T})$, $\boldsymbol{s} = \langle (s_1, t_1) \ \ldots \ (s_m, s_m) \rangle$ (*cf.* Définitions 1-2)
Output : $occs$: Ensemble d'occurrences

1 $adm \leftarrow \{[-\infty, \infty], \ldots, [-\infty, \infty]\}$ `// Région admissible de taille` m
2 $occs \leftarrow \emptyset$;
3 $\gamma_i \leftarrow$ `next`$(\gamma, \mathcal{C}, \boldsymbol{s})$ `// Choix du premier événement`
4 **foreach** $(s, t) \in \boldsymbol{s}$ **do**
5 **if** $s = e_{\gamma_1}$ **then**
  `// Création d'une nouvelle occurrence`
6 $o \leftarrow adm$;
7 $o_{\gamma_1} \leftarrow [t, t]$;
  `// Propagation des contraintes temporelles`
8 **foreach** $(e_{\gamma_1}, \gamma_1)[t^-, t^+](e, p) \in \mathcal{T}$ **do**
9 $o_p = [\max(0, t + t^-), \min(t + t^+, |\boldsymbol{s}| - 1)]$;
10 **foreach** $(e, p)[t^-, t^+](e_{\gamma_1}, \gamma_1) \in \mathcal{T}$ **do**
11 $o_p = [\max(0, t - t^+), \min(t - t^-, |\boldsymbol{s}| - 1)]$;
12 $loccs \leftarrow$ `RecRecognition`$(o, \gamma, 2, \mathcal{C}, \boldsymbol{s})$;
13 $loccs \leftarrow occs \cup loccs$;
14 **return** $occs$

contraintes temporelles de la chroniques concernant e_{γ_1} pour restreindre les régions admissibles. La ligne 11 réalise ensuite un appel récursif pour poursuivre l'exploration du multi-ensemble de la chronique. Cette fonction retourne la liste des occurrences de la chronique qui associe l'événement γ_1 du multi-ensemble avec la date t dans la séquence.

L'algorithme 2 présente la fonction récursive d'énumération des occurrences de la chronique. Les préconditions de la fonction sont que la région admissible en entrée contient des intervalles singletons pour tous les événements aux positions $\gamma_{1..(k-1)}$. La fonction effectue la recherche des positions pour les type d'événements c_{γ_k} dans la région admissible correspondante. Si de tels événements sont trouvés (ligne 7), alors la fonction propage les contraintes temporelles de la chronique qui impliquent e_{γ_k} (lignes 11 à 24) et poursuit la recherche d'occurrences jusqu'à avoir exploré tous les éléments du multi-ensemble (ligne 1 à 3) ou que les régions admissibles soient vides.

La fonction `next` représente l'heuristique de recherche. L'heuristique la plus simple est celle qui consiste à prendre tous les événements dans l'ordre du multi-ensemble. La seconde heuristique consiste à utiliser prioritairement les types d'événements qui apparaissent le moins dans la séquence. La troisième heuristique consiste à prioriser le traitement des événements du multi-ensemble qui sont les plus contraints en tenant compte de la fréquence d'apparition des événements dans la séquence, mais également la taille de la région d'admissibilité, suivant l'équation : $\operatorname{argmin}_{j \in [n]} f(e_j) \times (r_j^+ - r_j^-)$ où $f(e_j)$ est le nombre d'occurrences du type d'événements $e_j \in \mathcal{E}$ dans la séquence et $[r_j^-, r_j^+]$ est la région admissible pour l'événement $j \in [n]$ de la chronique.

4 Énumération des occurrences approchées d'une chronique

L'énumération des occurrences approchées d'une chronique consiste à énumérer toutes les (σ, λ)-occurrence d'une chronique pour des valeurs de σ et λ données.

Le principe de l'algorithme est très similaire à l'algorithme 1 pour lequel une occurrence est construite progressivement. Le calcul de la similarité entre une occurrence et la chronique

Algorithme 2 : `RecRecognition`$(o, k, \mathcal{C}, \boldsymbol{s})$.

Input : adm : Région admissible, γ : ordre d'exploration, k : niveau de récursion, $\mathcal{C} = (\mathcal{E} = \{\!\{e_1, \ldots, e_n\}\!\}, \mathcal{T})$, $\boldsymbol{s} = \langle (s_1, t_1) \ \ldots \ (s_m, s_m) \rangle$ (*cf.* Définitions 1-2)
Output : $occs$

```
1  if k = m + 1 then
2      return adm // Cas terminal
3  occs ← ∅ // Ensemble d'occurrences
4  γ_k ← next(γ, C, s);
5  foreach (s, t) ∈ s tel que t ∈ adm_k do
6      if s = e_{γ_k} then
           // Création d'une nouvelle occurrence
7          o ← adm;
8          o_{γ_k} ← [t, t];
           // Propagation des contraintes temporelles
9          satisfiable ← true;
10         foreach (e_{γ_k}, γ_k)[t⁻, t⁺](e, p) ∈ T, p ∉ γ_{1..(k−1)} do
11             o_p ← [max(o_p⁻, t + t⁻), min(o_p⁺, t + t⁺)];
12             if o_p⁻ > o_p⁺ then
13                 goto 6;
14         foreach (e, p)[t⁻, t⁺](e_{γ_k}, γ_k) ∈ T, p ∉ γ_{1..(k−1)} do
15             o_p ← [max(o_p⁻, t − t⁺), min(o_p⁺, t − t⁻)];
16             if o_p⁻ > o_p⁺ then
17                 goto 6;
18         loccs ← RecRecognition(o, γ, k + 1, C, s);
19         loccs ← occs ∪ loccs;
20 return occs
```

(*cf.* Définition 3) est réalisé également récursivement et dès que cette mesure est inférieure à la valeur σ, alors la région admissible courante est invalidée.

Pour réutiliser la stratégie d'exploration de l'algorithme 1, il est nécessaire de définir les limites d'exploration pour la recherche d'un événement de la chronique (région admissible). Dans le cas de la reconnaissance exacte d'une chronique, dès que l'événement sort de l'intervalle défini par les contraintes, il est écarté. Dans le cas de la reconnaissance approchée, une approche naïve nécessiterait d'explorer systématiquement l'intégralité de la séquence pour évaluer ς, la similarité entre une contrainte temporelle et l'occurrence d'un événement.

Le Lemme 2 nous indique que l'exploration exhaustive n'est pas nécessaire. Seules les régions définies par la chronique relaxée $R_{-\frac{1}{\lambda}\ln(\sigma)}(\mathcal{C})$ sont nécessaires. L'algorithme de reconnaissance approchée utilise ainsi les bornes de cette chronique pour affiner les régions admissibles à chaque étape de la reconnaissance (ligne 12 et 13 de l'algorithme 2). Lorsqu'une occurrence de $R_{-\frac{1}{\lambda}\ln(\sigma)}(\mathcal{C})$ est trouvée, alors la similarité est comparée à σ pour valider ou non l'occurrence comme une (σ, λ)-occurrence de $\mathcal{C}$.

5 Expérimentations

Dans cette section, on présente les résultats d'efficacité en temps de calcul obtenus sur des données synthétiques. L'utilisation de données synthétiques permet de mener des expérimentations contrôlées et pouvant représenter un large panel de l'utilisation possible de l'énumération de chroniques. L'algorithme a été implémenté en Python. [1]

1. Dépôt PIP : `https://pypi.org/project/pychronicles/` et Git : `https://gitlab.inria.fr/tguyet/pychronicles`

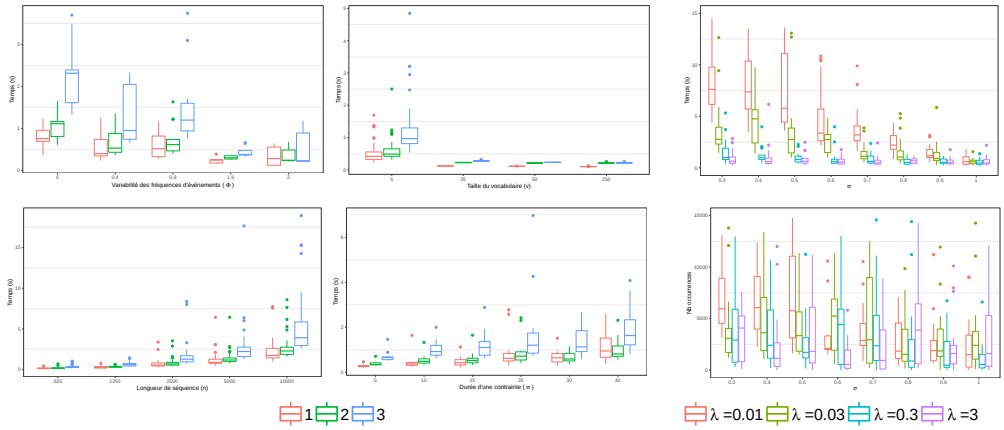

FIG. 1 – *Colonnes de gauche : Comparaison des temps de calcul entre les trois heuristiques (1 : ordre du multi-ensemble, 2 : ordre de fréquence, 3 : probabilité d'apparition) en fonction de différentes caractéristiques : en haut à gauche, de la variabilité des fréquences des événements (ϕ) ; au haut à droite, de la taille du vocabulaire (v) ; en bas à gauche, de la longueur des séquences (n) ; en bas à droite, de la durée des contraintes d'une chronique ($\sigma_{\mathcal{T}}$). Colonne de droite : Temps de calcul (en haut) et nombre de (σ, λ)-occurrences (en bas) en fonction de σ et de λ.*

Une séquence est générée comme une suite aléatoire de m types d'événements dans un vocabulaire de taille v ($v = 5$ et $m = 2000$ par défaut). Le tirage aléatoire des types d'événement se fait selon une loi multinomiale. La probabilité du i-ème type est donnée par $\frac{1}{m} + \phi \times \frac{i - \frac{m-1}{2}}{m \times (m-1)}$ où $\phi \in [0, 2]$ est un paramètre qui quantifie l'uniformité du tirage des types. Les dates des événements de la séquence sont générées en respectant une distribution normale $\mathcal{N}(\mu_{\mathcal{T}}, \sigma_{\mathcal{T}})$ des délais entre événements successifs ($\mu_{\mathcal{T}} = 4$ et $\sigma_{\mathcal{T}} = 1$ par défaut). Le multi-ensemble d'une chronique de taille n est généré comme un tirage aléatoire selon une loi multinomiale équiprobable sur le vocabulaire ($n = 4$ par défaut). La borne inférieure d'une contrainte temporelle est générée selon une loi normale $\mathcal{N}(\mu_{\mathcal{T}}, \sigma_{\mathcal{T}})$, comme pour les délais entre événements et la durée est obtenue comme la valeur absolue d'un tirage selon la loi $\mathcal{N}(0, 5 \times \mu_{\mathcal{T}})$. Le paramètre $\delta \in [0, 1]$ donne la probabilité d'avoir une contrainte temporelle entre deux événements (loi uniforme). On a ainsi en moyenne $\delta \times n \times (n-1)$ contraintes dans une chronique ($\delta = 0.4$ par défaut).

5.1 Comparaison des stratégies

La Figure 1 (2 colonnes de gauche) présente les temps de calcul obtenus pour des énumérations exactes d'une chronique aléatoire dans 100 séquences aléatoires. On observe d'une part que les temps de calcul sont assez faibles. Énumérer une chronique dans une séquence de 2000 événements prend moins de $0.01s$. D'autre part, on peut constater que l'heuristique naïve se montre plus performante que les deux autres.

Le graphique en haut à gauche illustre les temps de calcul en fonction de ϕ. On constate que la première heuristique est plus performante que les deux autres pour toutes les valeurs de $\phi < 2$. Plus ϕ est grand et plus les écarts de temps d'exécution sont petits, et c'est uniquement dans le cas où $\phi = 2$ que les seconde et troisième heuristiques se montrent plus performantes. Une explication possible à cette observation est que le temps nécessaire pour calculer les fréquences n'est pas compensé par la réduction du nombre de régions admissibles.

À partir de contraintes temporelles générées avec $\sigma_{\mathcal{T}} = 30$, la seconde heuristique devient plus performante que la première. C'est une situation où peu de régions admissibles sont éliminées du fait des contraintes temporelles (assez faibles) et, dans ce cas, la stratégie consistant à limiter leur génération en s'appuyant sur la fréquence des événements devient plus efficace.

Les deux autres graphiques montrent des comportements attendus de la reconnaissance de chroniques. Les temps de calcul décroissent très rapidement avec le taille du vocabulaire (graphique en haut à gauche). En revanche, les temps de calcul augmentent avec la taille des séquences (graphique en bas à gauche).

5.2 Approche exacte vs. approche approchée

Dans cette section, on étude l'impact des valeurs de σ et λ sur les (σ, λ)-occurrences d'une chronique. On utilise l'heuristique basée sur l'ordre des événements dans le multi-ensemble.

La Figure 1 (troisième colonne en haut) montre que les temps de calcul augmentent lorsque σ diminue. Plus λ est faible et plus cette augmentation est rapide. Ce comportement correspond à l'augmentation de la taille des régions admissibles de $R_{-\frac{1}{\lambda}\ln(\sigma)}(\mathcal{C})$. Plus les régions admissibles sont étendues et plus il faut parcourir la séquence à la recherche d'occurrences et donc plus les temps de calcul sont importants.

Les temps de calcul sont partiellement liés au nombre d'occurrences (Figure 1, en bas). On le constate bien sur l'évolution du nombre d'occurrences pour $\lambda = 0.01$: pour $\sigma = 1$ à 0.5, l'évolution du temps de calcul suit celui du nombre d'occurrences : plus on relaxe la contrainte de proximité et plus il y a d'occurrences. Passé un certain seuil, toutes les occurrences du multi-ensemble sont reconnues comme (σ, λ)-occurrence et le nombre d'occurrences n'augmente plus, bien qu'on élargisse les régions de recherche (et donc les temps de calcul).

6 Conclusion

Nous avons introduit un algorithme pour l'énumération d'occurrences exactes d'une chronique et introduit les (σ, λ)-occurrences qui sont des occurrences admettant que les dates ne respectent pas exactement les contraintes temporelles d'une chronique. L'algorithme d'énumération exacte peut alors être utilisé pour identifier efficacement les (σ, λ)-occurrences.

Nous avons expérimentalement évalué trois différentes heuristiques pour l'algorithme d'énumération et constaté que l'heuristique la plus simple se montre la plus efficace (sauf configuration extrême). Les améliorations du parcours par les heuristiques exploitant les distributions des types d'événements semblent ne pas compenser le temps de calcul nécessaire pour calculer ces distributions. Les expérimentations sur l'énumération des occurrences approchées d'une chronique montrent que cette tâche peut être réalisée avec des temps de calcul au plus supérieurs d'un ordre de grandeur pour $\lambda \approx 0.1$ et qui augmentent linéairement avec σ, le seuil de reconnaissance.

Financements

Ce travail a été partiellement financé par INTERREG Upper Rhine (Fond Européen de Développement Régional), les ministères de la recherche de Baden-Württemberg, Rheinland-Pfalz (Allemagne) et de la Région Grand Est dans le cadre du projet HALFBACK.

Références

Alur, R. et D. L. Dill (1994). A theory of timed automata. *Theoretical computer science 126*(2), 183–235.

Artikis, A., A. Skarlatidis, F. Portet, et G. Paliouras (2012). Logic-based event recognition. *The Knowledge Engineering Review 27*(4), 469–506.

Barazzutti, P.-L., A. Cordier, et B. Fuchs (2016). Transmute : un outil interactif pour assister l'extraction de connaissances à partir de traces. In *Actes de la conférence Extraction et Gestion des Connaissances (EGC)*, pp. 463–468.

Besnard, P. et T. Guyet (2019). *Chronicles*. à paraître.

Carle, P., C. Choppy, et R. Kervarc (2011). Behaviour recognition using chronicles. In *Proceedings of the International Conference on Theoretical Aspects of Software Engineering*, pp. 100–107.

Dechter, R., I. Meiri, et J. Pearl (1991). Temporal constraint networks. *Artificial Intelligence 49*(1-3), 61–95.

Ghallab, M. (1996). On chronicles : Representation, on-line recognition and learning. In L. Aiello, J. Doyle, et S. Shapiro (Eds.), *Proceedings of the 5th International Conference on Principles of Knowledge Representation and Reasoning (KR)*, pp. 597–606.

Kostakis, O. et P. Papapetrou (2017). On searching and indexing sequences of temporal intervals. *Data mining and knowledge discovery 31*(3), 809–850.

Monroe, M., R. Lan, J. Morales del Olmo, B. Shneiderman, C. Plaisant, et J. Millstein (2013). The challenges of specifying intervals and absences in temporal queries : A graphical language approach. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2349–2358. ACM.

Ulus, D. et O. Maler (2018). Specifying timed patterns using temporal logic. In M. Prandini et J. Deshmukh (Eds.), *Proceedings of the 21st International Conference on Hybrid Systems : Computation and Control (HSCC)*, pp. 167–176.

Summary

A chronicle is a set of events related by temporal constraints. It represents the behavior of a dynamic system. This temporal model has been studied in the context of monitoring and pattern mining. In this article, a chronicle is used to query a static database of sequences. We are interested in enumerating the exact or approximate occurrences of a chronicle. We propose a chronicle recognition algorithm with different heuristics and its version for approximate occurrences. The efficiency of the algorithm is evaluated on synthetic data.